SYSTEM AND METHOD FOR MANAGING SECURITY EVENTS ON A NETWORK

PRIORITY AND RELATED APPLICATIONS

The present application claims priority to provisional patent application entitled, "Event Manager for Network Security System," filed on April 28, 2000 and assigned U.S. Application Serial Number 60/200,313. The present application is also related to non-provisional application entitled, "Method and System for Managing Computer Security Information," filed on April 27, 2001, with attorney docket number 05456.105006 and to non-provisional application entitled "Method and System for Creating a Record for One or More Computer Security Incidents," filed on October 10, 2000 and assigned U.S. Application Serial Number 09/685,285.

TECHNICAL FIELD

The present invention is generally directed to managing security events on a network. More specifically, the present invention facilitates the collection and analysis of large amounts of security event data received from security devices for a distributed computer network.

BACKGROUND OF THE INVENTION

The security of computing networks is an increasingly important issue. With the growth of the Internet and the World Wide Web, people rely on computing networks to transfer and store more and more valuable information. This is particularly true in the context of local area networks (LANs) used by companies, schools, organizations, and other enterprises. LANs are used by groups of people to communicate and store documents and information. In the past, the security of computing networks was monitored by security devices placed on the network. Security devices generally comprise a sensor operable for generating a signal when it detects a certain event on the computing network. Security devices can also comprise components for managing and analyzing the data.

As more people began using computing networks more frequently, the networks have grown tremendously. With the growth in the size of networks and the importance of information available on the networks, there is a corresponding need for greater security monitoring. One approach to fulfilling this need has been to use a greater number and variety of security devices

to adequately monitor events taking place on the network. However, the use of more security devices to monitor networks creates more data for a monitoring system to handle. More security devices also result in more data for a user to review.

The current approach to monitoring networks with many security devices involves the use of consoles that can receive data from a group of security devices. However, there are many limitations with using existing consoles to monitor a large network. Generally, consoles are inherently limited in that they can only accept data from a few security devices. A further limitation is that users can only review and process data at a relatively slow rate. Finally, because each console on the network only receives information from a limited number of security devices, it is difficult to examine security data on a network-wide basis.

In view of the foregoing, there is a need in the art for a system which will support the collection of relatively large amounts of security event data from a network. Specifically, a need exists to be able to store, filter, and analyze the large amount of security event data so that it can be easily reviewed and managed by users monitoring the network. A further need exists to be able to customize the criteria for filtering the data. There is also a need to be able to collect and format data from a variety of different security devices located on a network. A further need exists to be able to analyze the collected data on a network-wide basis. Finally, there is a need to display the data in a simple graphical format for the users monitoring the network.

SUMMARY OF THE INVENTION

The present invention satisfies the above-described needs by providing a system and method for collecting security event data from security devices located in a distributed computing environment. The present invention improves upon existing approaches by providing a system that can collect, store, filter, and analyze security event data in order to facilitate managing the security for a relatively large computing network. A user can create customized scopes of varying criteria for filtering the data so that only the desired information is provided to a user. Scopes can also be customized to analyze security event data for responding to or anticipating a security event. By storing the security event data, the invention supports the retrieval of additional information about each event if needed. Improving the ability to manage security event data from a network further supports the capacity to respond to a security event when necessary.

In one aspect, the present invention comprises a system for managing security event data collected from a distributed computing network. The invention can include multiple security devices located throughout the network that generate security event data and a database server operable for collecting and storing the security event data. The invention can further comprise software modules operable for filtering and analyzing the security event data to produce resulting data for a client. In response to particular security event data, the client can create an incident report.

In another aspect, the present invention provides a method for managing a large amount of security event data collected from security devices comprising the steps of creating criteria for filtering and analyzing security event data, collecting security event data, and applying the criteria to the collected data to produce a result. The invention can accept and store the results produced from applying the criteria, and provide them to users of the event manager. A database server can also store the collected security event data and the criteria for later use. The results from applying the criteria can be rendered in a variety of different graphical formats including, but not limited to, tables, graphs, charts, and tree diagrams. The invention further supports additional analysis of the results and the creation of an incident report used in responding to the security event. Being able to process a large amount of data describing the security of an entire network enhances the ability to respond to a security event.

For yet another aspect, the present invention further provides a method for rendering selected resulting data from a large amount of security event data in a manageable format. The invention can comprise the steps of creating criteria operable for filtering and analyzing security event data, collecting security event data, applying the filtering and analyzing criteria to the security event data, and rendering the resulting data for a user. The method can support a variety of ways for rendering the analyzed data including, but not limited to, tables, graphs, charts, and tree diagrams. A database server can store the collected data, the criteria used for analyzing the data, and the results of the analysis. This inventive aspect further provides the ability to collect and analyze current security event data which allows for a timely response.

These and other aspects of the invention will be described below in connection with the drawing set and the appended specification and claim set.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram illustrating an exemplary operating environment for the present invention.

FIG. 2 is a block diagram illustrating an exemplary security event manager system.

FIG. 3 is a logic flow diagram illustrating an overview of the operating steps performed by an event manager in accordance with an exemplary embodiment of the present invention.

FIG. 4 is a logic flow diagram illustrating an exemplary process for opening a scope within the security event manager.

FIG. 5 is a logic flow diagram illustrating an exemplary process for creating a scope within the security event manager.

FIG. 6 is a logic flow diagram illustrating an exemplary process for editing a scope within the security event manager.

FIG. 7 is a logic flow diagram illustrating an exemplary process for deleting a scope within the security event manager.

FIG. 8 is a logic flow diagram illustrating an exemplary process for collecting data with the security event manager.

FIG. 9 is a logic flow diagram illustrating an exemplary process for analyzing data with the security event manager.

FIG. 10 is a logic flow diagram illustrating an exemplary process for polling for data within the security event manager.

FIG. 11 is a logic flow diagram illustrating an exemplary process for polling for messages within the security event manager.

FIG. 12 is a logic flow diagram illustrating an exemplary process for requesting event details from within the security event manager.

FIG. 13 is a logic flow diagram illustrating an exemplary process for clearing an event from the security event manager.

FIG. 14 is a logic flow diagram illustrating an exemplary process for creating an incident within the security event manager.

FIG. 15 illustrates an exemplary display screen of a main application window for the event manager.

FIG. 16 illustrates an exemplary display screen for sorting events in a table view.

FIG. 17 illustrates an exemplary display screen for clearing security event data from a table.

FIG. 18 illustrates an exemplary display screen for configuring a scope.

FIG. 19 illustrates an exemplary display screen for configuring a host group.

FIG. 20 illustrates an exemplary display screen for configuring a group of event types.

FIG. 21 illustrates an exemplary display screen for security event details.

DETAILED DESCRIPTION OF THE EXEMPLARY EMBODIMENTS

The present invention supports the management of security event data collected from a computing network. Specifically, the present invention allows large amounts of data in varying formats to be collected, stored, filtered, and analyzed according to selected criteria. A user can choose to filter the security event data so that only desired information is analyzed and rendered for monitoring. The ability to filter data provides the users with summaries of only the most important data and allows for greater amounts of information to be collected. If necessary, a user can retrieve more detail about a particular security event from the collected data that is stored. The ability to manage and analyze greater amounts of security event data provides for more effective monitoring of and response to security events.

Although the exemplary embodiments will be generally described in the context of software modules running in a distributed computing environment, those skilled in the art will recognize that the present invention also can be implemented in conjunction with other program modules for other types of computers. In a distributed computing environment, program modules may be physically located in different local and remote memory storage devices. Execution of the program modules may occur locally in a stand-alone manner or remotely in a client/server manner. Examples of such distributed computing environments include local area networks of an office, enterprise-wide computer networks, and the global Internet.

The detailed description which follows is represented largely in terms of processes and symbolic representations of operations in a distributed computing environment by conventional computer components, including database servers, application servers, mail servers, routers, security devices, firewalls, clients, workstations, memory storage devices, display devices and

input devices. Each of these conventional distributed computing components is accessible via a communications network, such as a wide area network or local area network.

The processes and operations performed by the computer include the manipulation of signals by a client or server and the maintenance of these signals within data structures resident in one or more of the local or remote memory storage devices. Such data structures impose a physical organization upon the collection of data stored within a memory storage device and represent specific electrical or magnetic elements. These symbolic representations are the means used by those skilled in the art of computer programming and computer construction to most effectively convey teachings and discoveries to others skilled in the art.

The present invention also includes a computer program which embodies the functions described herein and illustrated in the appended flow charts. However, it should be apparent that there could be many different ways of implementing the invention in computer programming, and the invention should not be construed as limited to any one set of computer program instructions. Further, a skilled programmer would be able to write such a computer program to implement the disclosed invention based on the flow charts and associated description in the application text, for example. Therefore, disclosure of a particular set of program code instructions is not considered necessary for an adequate understanding of how to make and use the invention. The inventive functionality of the claimed computer program will be explained in more detail in the following description in conjunction with the remaining figures illustrating the program flow.

Referring now to the drawings, in which like numerals represent like elements throughout the several figures, aspects of the present invention and the preferred operating environment will be described.

FIG. **1** illustrates various aspects of an exemplary computing environment in which the present invention is designed to operate. Those skilled in the art will appreciate that FIG. **1** and the associated discussion are intended to provide a brief, general description of the computer network resources in a representative distributed computer environment including security devices and the inventive event manager.

Referring to FIG. **1**, an exemplary operating environment for the present invention is illustrated. FIG. **1** shows a local area network (LAN) **105** such as that found in a typical company. The LAN **105** can comprise a workstation **110** and a client **115** coupled to the

network. A typical LAN **105** can support multiple workstations and clients. The LAN **105** can further comprise a firewall **125** for controlling the flow of electronic data into and out of the network. A router **120** manages the flow of electronic information and data within the network and beyond the network to ensure that packets of electronic data arrive at the correct location. An exemplary LAN **105** can also comprise a mail server **130**.

In order to monitor the security of the network, a typical LAN will have numerous security devices located throughout the network. In the exemplary LAN **105** illustrated in FIG. **1**, the security devices are represented as a single element **135**, but are not limited to this configuration. In other embodiments of the present invention security devices can be located at various points throughout the network or installed or embedded within other systems. An event manager **140**, comprising a database server **145** and an application server **150**, can also be coupled to the LAN **105**. The event manager **140** is operable for collecting security event data from the security devices **135** as described in more detail below in connection with FIG. 2. The local area network **105** can also be coupled to a wide area network **160** such as the World Wide Web. A connection to a wide area network **160** enables remote clients **165** and **170** to access the local area network **105**. A typical LAN **105** will also have a firewall **155** between the LAN **105** and the wide area network **160**. Although the event manager **140** is described in this representative environment as operating on a LAN, it should be understood that the invention can operate on a variety of distributed computing networks.

Referring to FIG. **2**, an exemplary architecture for the event manager **140** is illustrated. The event manager **140** facilitates the management of large amounts of security event data by gathering the data, automatically analyzing the data, and providing the results of any analysis to the users of the event manager **140**. As shown in FIG. **1**, the event manager **140** comprises the database server **145** and the application server **150**. The event manager **140** is coupled to security devices **135** illustrated as discrete security systems **205**, **210**, and **215**. Security event data from the security systems **205**, **210**, and **215** is gathered by the collector **225** on the database server **145**. The database server **145** can further comprise a database **220**, operable for storing data, and an analyzer storage module **230**, operable for storing procedures for analyzing security event data. The application server **150** is typically coupled to a client **115** and the local area network **105**. Although not shown in FIG. **2**, a remote client **165** can also access the application

server **150**. The application server **150** can comprise numerous software modules for managing the security event data collected from the security devices **135**. The application server **150** can comprise an analyzer module **265** for analyzing security event data collected by the security devices **135**. The application server **150** can also comprise a persistence module **245** for managing the storage of data and a garbage collector module **270** for disposing of unneeded data. The result module **235** provides client **115** with results from the analyzer module **265**. The client **115** can also get additional data concerning a security event from the database server **145** through the event details module **250**. If a person using client **115** sees an event of significance, an incident can be created with the incident response module **255**. A message module **260** supports the collection of messages for a client **115** concerning activities performed on the event manager **140**.

FIG. **3** illustrates an overview of the exemplary processes that the event manager **140** performs. It should be understood that the steps described throughout the invention description can be performed automatically by software modules operating in conjunction with the event manager **140** or by a person operating a client coupled to the event manager **140**. Steps **305** and **310** are threshold steps that typically are performed before the event manager **140** performs any analysis. In step **305**, either a software module or a person operating client **115** can set up various kinds of scopes for filtering, analyzing, and rendering event data collected by the security devices **135**. Different sets of criteria for filtering and analyzing the data can be employed in each scope and the particular scopes can be stored for subsequent use by clients. Throughout the description of the invention, the term "scope" can encompass filtering or analyzing processes, or a combination of the two processes. Furthermore, the term analyzing can include both filtering and analyzing of data. Step **310** illustrates the collection of data from the security devices **135**. The collector **225** on database server **145** performs the data collection illustrated in step **310**. Step **315** illustrates the analysis performed by the analyzer module **265** on the application server **150**. The procedures run by the analyzer module **265** are typically stored on the database server in the analyzer storage module **230**. Additionally, preliminary analysis of the data can occur at the security devices **135**. Further information concerning preliminary filtering and analysis of data is contained in the non-provisional application entitled, "Method and System for Managing Computer Security Information," filed on April 27, 2001, with attorney docket number 05456.105006.

In step **320**, client **115** can monitor the event data using the selected scopes. The monitoring of data can include further analysis of the data, polling for messages concerning the data, and requesting additional details about particular security events. In step **325**, client **115** can clear event data that has previously been stored. For example, event data pertaining to an event that is no longer deemed to be significant can be cleared from storage. In step **330**, client **115** can create an incident for a particular event that is deemed to be significant. By creating an incident, the event data is stored by the event manager **140** for future analysis or response. In alternative embodiments of the present invention, the processes performed in steps **325** and **330** can be performed automatically by the event manager **140** or incorporated into the analysis step **315**. The ability to automate steps shown in FIG. 3 further supports the management of large amounts of security event data. Additionally, the order of the foregoing steps exemplifies a typical event manager. Alternative embodiments of the present invention may combine certain steps or perform them in a different order.

FIGs. **4**, **5**, **6**, and **7** illustrate exemplary processes for set-up and use of the scopes that filter and analyze security event data. FIG. **4** illustrates an exemplary process for a client to open a scope. In step **405**, client **115** requests the configuration for a desired scope from the application server **150**. The configuration criteria for the scopes can be stored on the analyzer storage module **230** of database server **145**. Typical configuration criteria include sorting security event data by destination address or event type. In step **410**, the persistence module **245** retrieves the configuration criteria for the desired scope from the database server **145**. In step **415**, client **115** opens the desired scope. In step **420**, client **115** can initialize and render the display for the scope on an output device, such as a monitor or printer. The display for the scope can comprise one or more tables, charts, graphs, tree diagrams, or other renderings for presenting data to a user.

FIG. **5** illustrates an exemplary process supported by a user to create a new scope for filtering or analyzing data. In step **505**, a software module operating within the event manager **140** or a user operating client **115** can define the filter and/or analysis for a new scope. In defining a scope, the user or software module selects certain criteria for filtering and/or analyzing security events, such as a source or destination address, an event type, or the type of sensor. The present invention supports a variety of criteria for filtering and analyzing security

event data. In step **510**, the analyzer module **265** converts the selected criteria to a scope definition. The scope definition describes what filtering and analysis will be performed on the security event data. In step **515**, the persistence module **245** stores the scope definition. The analyzer module **265** then converts the scope definition into a stored procedure in step **520**. The stored procedure is a list of instructions that execute when supplied with security event data for the corresponding scope. The persistence module **245** stores the procedure on the database server **145** in step **525**. An alternative embodiment can support creating a record of all procedures performed by the event manager **140**. Thus, in step **530**, when the scope is created, a confirmation message is sent to the client by the analyzer module **265** and/or stored in the message module **260** on the application server **150**.

The foregoing steps are merely an exemplary means for presenting the security event data in a graphical and/or text-based format. It should be evident to those skilled in the art that other procedures may be used for rendering the data on an output device. Furthermore, although the creation of a scope is illustrated in step **305** of FIG. **3**, a user can create a new scope at any step in FIG. **3**.

FIG. **6** illustrates an exemplary process for editing an existing scope. The process illustrated in FIG. **6** can be used to change the filtering or analyzing criteria of a particular scope. In step **605**, a software module operating on the event manager **140** or the client **115** can request the scope definition for a particular scope to be edited. In step **610**, the persistence module **245** retrieves the stored scope definition from the database server **145**. In step **615**, the analyzer module **265** converts the scope definition to the originally selected filtering and analyzing criteria. The filtering and analyzing definition is modified by the client **115** in step **620** by deleting and/or adding criteria. The modified scope is then stored by the persistence module **245** in the analyzer storage module **230** according to steps **510** through **530** of FIG. **5**. In step **625**, the event manager **140** confirms the editing of the scope by sending an electronic message to the message module **260**.

Rather than edit a scope, if a user wishes to delete a scope entirely, FIG. **7** illustrates an exemplary process for doing so. The exemplary embodiment provides a user with a particular option for deleting a scope. In step **705**, client **115** can send a scope delete request to the

persistence module **245**. In step **710**, the persistence module **245** will remove the scope definition from the database **220** on the database server **145**. The stored procedure associated with the criteria for the selected scope is then removed from the database server **145** in step **715**. Client **115** will also receive a confirmation message that the scope has been deleted in step **720**. The foregoing methods illustrated in FIGs. **4**, **5**, **6** and **7** demonstrate that embodiments of the present invention can support a variety of filtering and analysis tasks to meet the user's needs.

The data collection step **310** of FIG. **3** is illustrated in greater detail in FIG. **8**. FIG. **8** illustrates an exemplary process for collecting data from security devices **135**. In step **805**, a sensor within a security system located on the network generates a security event. The data generated from this security event is sent to the collector **225** in step **810**. In an alternative embodiment of the present invention, some filtering or analysis of data may occur at the security system before data is forwarded to the collector **225**. Because the collector **225** is gathering data from a variety of different security systems located throughout the network, the collector **225** preferably converts the varied data to a uniform format. In step **815**, the collector **225** converts all the gathered event data to a common format. In step **820**, the event data, once converted to a common format, is stored in the database **220** for future use or analysis.

FIG. **9** illustrates in greater detail the analysis step **305** of FIG. **3** by showing an exemplary process for an event manager **140** to analyze security event data. The analysis of data may be initiated by a scheduled trigger within the event manager as in step **905** or in response to an external request from a user as in step **910**. An analysis of data typically occurs over a defined time period. In step **915**, client **115** inputs a particular start time or the scheduled start time is sent to the analyzer module **265**. The event manager **140** supports a variety of analyses to be performed on the collected security event data. These analyses are stored as procedures in the analyzer storage module **230** on the database server **145**. For example, an exemplary analysis procedure may compare the source address for security events detected by different security systems located throughout the network. Another exemplary analysis procedure may compare security events and known vulnerabilities for a particular network. The event manager's ability to perform these analyses on a much larger scale of data than traditional approaches supports more effective security monitoring and response to security events.

In step **920**, client **115** can invoke a stored analysis procedure. The analysis procedure executes for all event data collected between the start time and the current time in step **925**. If the analysis was initiated by a predetermined schedule, the results of the analysis are stored in the database **220** in step **935**. If the analysis was performed in response to an external trigger, the "Yes" branch is followed to step **940** where the results of the analysis are presented to client **115**. The results of the analysis are typically rendered for the client in a graphical user interface containing tables, charts, graphs, diagrams or other renderings. The three-tier architecture of the present invention enables more rigorous analyses to be performed on larger volumes of security event data than is capable with the traditional console approach. By automatically applying the analysis procedures to large collections of security event data, the event manager **140** allows for more effective network security management. The three-tier architecture also facilitates sharing of information among a plurality of clients being used to conduct security monitoring.

FIGs. **10**, **11** and **12** illustrate exemplary processes by which users can continue to monitor security event data as it is collected by the event manager **140**. FIG. **10** illustrates an exemplary process for polling data. In step **1005**, the client **115** sends a request for particular data to the results module **235**. In step **1010**, if the request corresponds to a scheduled analysis, the "No" branch is followed to step **1015** and the stored results for the analysis are retrieved by the persistence module **245** from the database **220**. If the request in step **1010** is externally triggered, the "Yes" branch is followed to step **1020** where steps **910** through **940** from FIG. 9 are performed. In step **1025**, the resulting data is added to a running list to be supplied to the client **115**. If there are more analyses in the request, the user returns to step **1010** and the foregoing process is repeated. If no more analyses are requested in step **1030**, the "No" branch is followed to step **1035** where the results are returned by the results module **235** to the client **115**. When the results are returned, the client **115** can choose from a variety of formats for rendering the results in step **1040**.

FIG. **11** illustrates an exemplary process whereby a client **115** can poll for electronic messages from the event manager **140**. In step **1105**, client **115** sends a request for messages to the results module **235**. In step **1110**, the results module **235** looks to see if there are any new messages for the client. If not, the "No" branch is followed to step **1115** and the client is informed that there are no new messages. If there are new messages, the "Yes" branch is

followed to step **1120** where the new messages are returned to the client. In step **1125**, client **115** can take the appropriate action based on the message. For example, if there is a message that a needed scope was created, client **115** can then access that scope using the event manager **140**.

Referring to FIG. **12**, an exemplary process is illustrated for requesting additional event details. Although the discussion and drawings refer to a single event, the invention can support the processing of data corresponding to multiple security events. When a user views a particular scope that is of interest, the user may request additional data concerning a security event. In step **1205**, client **115** will select the desired security event. In step **1210**, client **115** can choose the event details option provided by the event manager **140**. The client's request is sent to the event details module **250** in step **1215**. The event details module **250** will query the database **220** for the additional security event data in step **1220**. In step **1225**, the event details are provided to client **115**. The filtering and analysis functions of the event manager **140** are important in that they provide manageable summaries of data to the users of the event manager. However, an equally important feature is the ability of the event manager **140** to retrieve the additional security event data not shown in the summaries.

FIG. **13** illustrates in greater detail the exemplary clear event step **325** of FIG. **3**. The process begins in step **1305** when a client **115** selects an event. In step **1310**, the client **115** chooses the clear event option. A request is sent to the clear events module **240** in step **1315**. The clear events module **240** clears the selected event from the database server **145** in step **1320**. In step **1325**, a message confirming the clearing of the event is sent to client **115**. The clear events function of the event manager **140** aids in the management of data by purging unneeded data.

FIG. **14** illustrates in greater detail the process for creating an incident as referred to in step **330** of FIG. **3**. Further information concerning incident records of security events is contained in and fully incorporated herein by reference to the non-provisional application entitled "Method and System for Creating a Record for One or More Computer Security Incidents," filed on October 10, 2000, by the assignee of this application and assigned U.S. Application Serial Number 09/685,285. FIG. **14** illustrates an exemplary process whereby a user can create an incident for security event data that is deemed significant. In step **1405**, the client

115 selects event data that is believed to signify an important security event. In step **1410**, the client chooses the create incident option provided by the event manager **140**. In step **1415**, the client creates an incident definition with the relevant event data. The incident definition is then sent to the incident module **255** in step **1420**. In step **1425**, the incident module **255** stores the incident. The ability to create an incident allows a user to identify particular event data for a response team. After creating the incident, the user can continue to monitor security event data in real time. In step **1430**, a confirmation message is sent to the client **115**.

Referring to FIG. 15, an exemplary screen display is illustrated for the event manager **140**. FIG. 15 illustrates an exemplary main application window **1500** employed by a user monitoring a computing network with the event manager **140**. The main application window **1500** can comprise various other windows including a message window **1520** showing event manager activities. Window **1505** lists the scopes defined for the computing network being monitored. A user can select one of the scopes listed in window **1505** in order to view filtered security event data in a chart **1510** or table **1515** format. The table **1515** typically indicates when a security event took place, the source and destination addresses of the security event, the event type and priority, and the system that detected the security event. Other types of graphical formats, such as tree diagrams, can also be used to present data. FIGs. 16 and 17 illustrate screen displays **1600** and **1700** of other exemplary tables. The data in the table can be sorted in a variety of ways. FIG. 16 shows event data sorted by priority. FIG. 17 depicts a user clearing security event data.

The event manager also facilitates the creation of new scope criteria for the user. FIG. 18 illustrates an exemplary screen display **1800** for configuring a scope. With this display, a user can name the scope, choose an interval for the scope to run, and select the criteria that define the scope. FIGs. 19 and 20 show examples of additional features that assist a user in creating scope criteria. FIG. 19 is an exemplary screen display **1900** showing the host group feature for grouping addresses in a distributed computing environment. Host groups can be used in creating criteria for defining a scope. FIG. 20 illustrates an exemplary screen display **2000** for grouping types of known security events. Filtering data by security event type can be useful way of managing the data.

FIG. 21 illustrates an exemplary screen display **2100** for presenting a user with additional event details. The event details window **2105** can provide additional information such as the source and destination ports for the security event.

In conclusion, the present invention enables and supports the management of large amounts of security event data collected from a computing network. The event manager can gather data from a variety of security devices, place the data in a uniform format, and store the data for later access. The invention allows a user to create criteria for filtering and analyzing the security event data so that manageable summaries of the data are presented to the user. The invention can also present the data to the user in a variety of formats. The ability to manage large amounts of security event data enables a user to more effectively monitor a computing network and respond to any security threats.

It will be appreciated that the present invention fulfills the needs of the prior art described herein and meets the above-stated objects. While there has been shown and described the preferred embodiment of the invention, it will be evident to those skilled in the art that various modifications and changes may be made thereto without departing from the spirit and the scope of the invention as set forth in the appended claims and equivalence thereof. Although the present invention has been described as operating on a local area network, it should be understood that the invention can be applied to other types of distributed computing environments. Furthermore, it should be readily apparent that the components of the event manager can be located in various local and remote locations of a distributed computing environment.